UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/713,343	11/14/2003	Claude Basso	RPS920030063US1	9860

45211 7590 12/24/2008

Robert A. Voigt, Jr.
WINSTEAD SECHREST & MINICK PC
PO BOX 50784
DALLAS, TX 75201

EXAMINER	
FEARER, MARK D	

ART UNIT	PAPER NUMBER
2443	

MAIL DATE	DELIVERY MODE
12/24/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/713,343	BASSO ET AL.	
	Examiner	Art Unit	
	MARK D. FEARER	2443	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *05 October 2008*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *2-5 and 7* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *2-5 and 7* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Applicant's Amendment filed 05 October 2008 is acknowledged.

2. Claims 2 and 7 have been amended.

3. Claims 6 and 20 are cancelled.

4. Claims 2-5 and 7 are still pending in the present application.

5. This action is made FINAL.

Claim Rejections - 35 USC § 103

6. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

The factual inquiries set forth in *Graham* **v.** *John Deere Co.*, 383 U.S. 1, 148 USPQ 459 (1966), that are applied for establishing a background for determining obviousness under 35 U.S.C. 103(a) are summarized as follows:

1. Determining the scope and contents of the prior art.
2. Ascertaining the differences between the prior art and the claims at issue.
3. Resolving the level of ordinary skill in the pertinent art.
4. Considering objective evidence present in the application indicating obviousness or nonobviousness.

This application currently names joint inventors. In considering patentability of the claims under 35 U.S.C. 103(a), the examiner presumes that the subject matter of the various

claims was commonly owned at the time any inventions covered therein were made absent any evidence to the contrary. Applicant is advised of the obligation under 37 CFR 1.56 to point out the inventor and invention dates of each claim that was not commonly owned at the time a later invention was made in order for the examiner to consider the applicability of 35 U.S.C. 103(c) and potential 35 U.S.C. 102(e), (f) or (g) prior art under 35 U.S.C. 103(a).

7. Claims 2-5 and 7 are rejected under 35 U.S.C. 103(a) as being unpatentable over Ramanathan et al. (US 6910063 B1) in view of Gupta et al. (US 6704786 B1) and in further view of Jindal et al. (US 6327622 B1).

Consider claim 2. Ramanathan et al. discloses a method for reducing the number of messages to be processed by a control processor in a load balancer comprising the steps of: establishing said TCP connection with said client via handshake messages between said network processor and said client ((“This problem exists for each Web server that utilizes and processes HTTP client requests, and is compounded with every hit that the Web server receives. For larger Web servers that utilize multiple processors to enable the servicing of many more transactions per second, this problem is multiplied. The performance of these multi-processor Web servers is further detrimentally impacted by the TCP/IP requirement for the queuing of kernel mode asynchronous procedure calls (k-mode APCs) for all network input/output (IO) requests (embodied in IO request packets or IRPs). As illustrated in FIG. 6, when a thread 532 running on one processor 534 of the multi-processor server generates network IO 536,

e.g. the generation of Send IRP in response to a client HTTP get request, a kernel-mode APC in the context of his thread 532 is scheduled to write a result of the IO upon its completion. In this example, the Send IRP is completed when the server receives an Ack for the last byte of data comprising the send from the client (see FIG. 5 for an illustration of the completion of Send IRP when Ack 516 is received from the client).") column 3 lines 32-51); receiving a request message from said client; bundling said request message and information from said handshake messages involved in establishing said TCP connection by said network processor; transmitting said bundled message to said control processor by said network ((“The methods of the instant invention increase the performance throughput of single and multiple processor network servers that service HTTP transactions. Within each individual server, a method is presented to enhance its performance during the processing of static non-keep alive HTTP transactions. This performance enhancement is achieved by bundling TDI_Send, TDI_Disconnect into a single IRP, and by handing notification with the completion of this signal IRP. Further performance is achieved in multiple processor servers by removing the queuing stage of the completion processing to complete the IRPs. As a result, these IRPs are completed directly, saving IPIs that otherwise would be generated as the TCP generated queue of these completions is drained.") column 4 lines 34-47); bundling said client's request message and a control message by said control processor; transmitting said bundled message comprising said client's request message and said control message to said network processor ((“The performance enhancements made available through the system of the invention described above all presuppose completion of the

IRP on the processor that originated it. While this is a foregone conclusion for single processor servers, conventional multiple-processor servers require a queuing state and an IPI to ensure that all IRPs are so completed as discussed above with reference to FIG. 7. However, this queuing state and subsequent IPI generation creates extra overhead for the server and adversely impacts its performance. Therefore, in multiple-processor servers employing the system of the instant invention, all IRPs are directly completed on the originating processor without a queuing state as illustrated in FIG. 3. As is illustrated, the processor 300 running thread 302 directly receives the IRP completion 304 without a queuing state as required by conventional systems. Other processors 306 in the server are not required to service this IRP completion 304, and therefore need generate no IPIs. The system of the invention also minimizes the cache line movement incurred conventional systems, which further enhances performance. A direct completion path is included in the system of the present invention for the bundled Send and Disconnect IRPs discussed above (see FIG. 2). Similar to the direct completion of the conventional Send IRPs and disconnect IRPs, the bundled Send and Disconnect IRPs are directly completed through the SendComplete path. This path occurs if the connection is gracefully closed sooner than NDIS's (Network Driver Interface Specification) SendComplete call for the last data frame. With current high throughputs of modern servers, this is a definite possibility especially in files that can fit into one jumbo data frame.") column 11 lines 25-56); receiving a request to terminate said TCP connection from said server by said network processor; facilitating said termination of said connection between said server and said client; bundling information

regarding a series of closed connections by said network processor, wherein said bundling information occurs after a particular number of connections have been closed; and transmitting said bundled message regarding said series of closed connections to said control processor by said network processor ((“The methods of the instant invention increase the performance throughput of single and multiple processor network servers that service HTTP transactions. Within each individual server, a method is presented to enhance its performance during the processing of static non-keep alive HTTP transactions. This performance enhancement is achieved by bundling TDI_Send, TDI_Disconnect into a single IRP, and by handing notification with the completion of this signal IRP. Further performance is achieved in multiple processor servers by removing the queuing stage of the completion processing to complete the IRPs. As a result, these IRPs are completed directly, saving IPIs that otherwise would be generated as the TCP generated queue of these completions is drained.”) column 4 lines 34-47 (“Since the server 502 knows that it may close the TCP connection once the requested file has been transferred to the client 500, the kernel mode provider 506 also generates a TdiDisconnect request 518 to the transport layer 508. The transport layer continues to send data frames to the client 500. The typical or average amount of data that is transmitted to the client 500 is approximately 14 kbytes of information. A "large" (or Jumbo) data frame may hold up to 8 kbytes of information, and therefore typically two data frames 514a and 514b are transmitted to the client 500 at a time. Upon the final transmission of the two large data frames, the transport layer also transmits a third data frame containing a FIN to signify to the client 500 that the server is closing the TCP

connection. Unfortunately, while this third data frame contains only the FIN message (and appropriate header information), its generation requires approximately the same amount of processor time in the server to generate as the data frames containing the actual requested information. Therefore, this third large data frame 520 is essentially empty, except for the FIN, and yet requires essentially the same amount of server processing to generate. This greatly increases the overhead of the server, and detrimentally impacts its performance. In response to the transmission of the last data frames 514a, 514b, and the FIN data frame 20, the client 500 transmits the transmission ack 516 to acknowledge the receipt of the data frames, and a FIN-ack 522 to acknowledge the receipt of the FIN message 520. In response to each of these ack's 516, 522, the transport layer 508 generates an Irp-completion 524, 526 to signify the completion of both the TdiSend 512 and the TdiDisconnect 518 respectively. Additionally, in response to the receipt of the FIN-ack 522, the transport layer notifies 528 the kernel mode provider 506 of the appropriateness of a graceful disconnect (client has acknowledged closing of connection from the server's end) with the client 500. As a result, the kernel mode provider closes the TCP connection and generates a notification 530 up to the user mode server application 504 that the TCP connection has been successfully, gracefully disconnected from the client. Unfortunately, the processing of each of the Irp-completion 524, 526 incurs hundreds of processor cycles of overhead. Therefore, the processing of multiple Irp-completions also has an impact on overall server performance. That is, for each HTTP Get request, two Irp-completions must be

processed, each of which requiring 30 hundred cycles of the server's processing capabilities.") column 2 line 41 – column 3 line 19).

However, Ramanathan et al. fails to disclose a method of receiving a request to establish a TCP connection from a client by a network processor in a load balancer.

Gupta et al. discloses a method of receiving a request to establish a TCP connection from a client by a network processor in a load balancer (("FIG. 2 illustrates a TCP data transfer use as a fall back in accordance with one aspect of the invention when a UDP data transfer cannot be affected. The vertical line to the left of FIG. 2 represents the client and the vertical line to the right represent the server who are engaged in interaction as indicated; time progresses from top to bottom in FIG. 2. The client sends a SYN packets (Synchronization Packet) to the server (200). This initiates a request that a connection be established from client to server. The server responds with a SYN-ACK packet which is a request for a connection in the other direction (from the server to the client) and an acknowledgment of the set-up of a connection from the client to the server at the server end (210). The client then responds with an ACK (220) which acknowledges to the server that the connection requested has been set-up. The interchange of these three packets constitutes a connection set-up phase at the end of which, a virtual circuit has been established in each direction between client and server and between server and client. The client then requests a page (document, data) from the server using the GET PAGE XXX request (230). The server sends an ACK

acknowledging the page request together with the data which responds to that request in one or more packets (240). Acknowledgments are generated as data packets are received in a manner specified in the TCP protocol. When the final acknowledgement is sent from the client to the server (250) the data transfer phase of the connection has been completed. With the last data packet acknowledged, the client sends a FIN packet (260) to the server indicating that the connection is finished. The server responds with a FIN-ACK packet which acknowledges the FIN packet of the client and indicates to the client that the use of the reverse direction connection is also finished (270). When the client acknowledges the FIN-ACK packet from the server (280), the connection is terminated and the connection tear down phase is completed.") column 6 lines 7-42).

Ramanathan et al. discloses a prior art system and method of enhancing web server throughput in single and multiple processor systems wherein establishing a TCP connection with a client via handshake messages between a network processor and a client; receiving a request message from said client; bundling said request message and information from said handshake messages involved in establishing said TCP connection by said network processor; transmitting said bundled message to a control processor by a network; bundling said client's request message and a control message by said control processor; transmitting said bundled message comprising said client's request message and said control message to said network processor; receiving a request to terminate said TCP connection from said server by said network processor; facilitating said termination of said connection between said server and said client;

bundling information regarding a series of closed connections by said network processor, wherein said bundling information occurs after a particular number of connections have been closed; and transmitting said bundled message regarding said series of closed connections to said control processor by said network processor upon which the claimed invention can be seen as an improvement.

Gupta et al. teaches a prior art comparable network and end-host efficiency for web communication comprising a method of receiving a request to establish a TCP connection from a client by a network processor in a load balancer.

Thus, the manner of enhancing a particular device (network and end-host efficiency for web communication comprising a method of receiving a request to establish a TCP connection from a client by a network processor in a load balancer) was made part of the ordinary capabilities of one skilled in the art based upon the teaching of such improvement in Gupta et al. Accordingly, one of ordinary skill in the art would have been capable of applying this known improvement technique in the same manner to the prior art system and method of enhancing web server throughput in single and multiple processor systems wherein establishing a TCP connection with a client via handshake messages between a network processor and a client; receiving a request message from said client; bundling said request message and information from said handshake messages involved in establishing said TCP connection by said network processor; transmitting said bundled message to a control processor by a

network; bundling said client's request message and a control message by said control processor; transmitting said bundled message comprising said client's request message and said control message to said network processor; receiving a request to terminate said TCP connection from said server by said network processor; facilitating said termination of said connection between said server and said client; bundling information regarding a series of closed connections by said network processor, wherein said bundling information occurs after a particular number of connections have been closed; and transmitting said bundled message regarding said series of closed connections to said control processor by said network processor of Ramanathan et al. and the results would have been predictable to one of ordinary skill in the art, namely, one skilled in the art would have readily recognized a system and method of web server throughput.

However, Ramanathan et al., as modified by Gupta et al., fails to disclose a method of identifying a server in a server farm to service a client's request message by a control processor.

Jindal et al. discloses a method of identifying a server in a server farm to service a client's request message by a control processor ((“Each server farm in the presently described embodiment includes an intermediate server (i.e., server 306 in server farm 300 and server 316 in server farm 310). One function of an intermediate server in this embodiment is to collect, from the servers in the farm that host instances of the load-balanced application information necessary to select a preferred server. For example,

intermediate replicated monitor object (IRMO) 306a is operated on intermediate server 306 to collect data from servers 302 and 304. IRMO 306a operates similarly to the RMO described above in conjunction with to FIG. 2, but in this embodiment is located on a server situated between central server 100 and the servers offering the application. The load balancing framework of the illustrated embodiment also includes status objects (e.g., depicted by numerals 302a, 304a, 312a and 314a) and IMOs (e.g., depicted by numerals 302b, 304b, 312b and 314b) operating on servers 302, 304, 312 and 314.") column 9 line 59 – column 10 line 8).

Ramanathan et al., as modified by Gupta et al., discloses a prior art system and method of enhancing web server throughput in single and multiple processor systems wherein establishing a TCP connection with a client via handshake messages between a network processor and a client; receiving a request message from said client; bundling said request message and information from said handshake messages involved in establishing said TCP connection by said network processor; transmitting said bundled message to a control processor by a network; bundling said client's request message and a control message by said control processor; transmitting said bundled message comprising said client's request message and said control message to said network processor; receiving a request to terminate said TCP connection from said server by said network processor; facilitating said termination of said connection between said server and said client; bundling information regarding a series of closed connections by said network processor, wherein said bundling information occurs after

a particular number of connections have been closed; and transmitting said bundled message regarding said series of closed connections to said control processor by said network processor; and_network and end-host efficiency for web communication comprising a method of receiving a request to establish a TCP connection from a client by a network processor in a load balancer upon which the claimed invention can be seen as an improvement.

Jindal et al. teaches a prior art method of identifying a server in a server farm to service a client's request message by a control processor.

Thus, the manner of enhancing a particular device (method of identifying a server in a server farm to service a client's request message by a control processor) was made part of the ordinary capabilities of one skilled in the art based upon the teaching of such improvement in Jindal et al. Accordingly, one of ordinary skill in the art would have been capable of applying this known improvement technique in the same manner to the prior art system and method of enhancing web server throughput in single and multiple processor systems wherein establishing a TCP connection with a client via handshake messages between a network processor and a client; receiving a request message from said client; bundling said request message and information from said handshake messages involved in establishing said TCP connection by said network processor; transmitting said bundled message to a control processor by a network; bundling said client's request message and a control message by said control processor; transmitting

said bundled message comprising said client's request message and said control message to said network processor; receiving a request to terminate said TCP connection from said server by said network processor; facilitating said termination of said connection between said server and said client; bundling information regarding a series of closed connections by said network processor, wherein said bundling information occurs after a particular number of connections have been closed; and transmitting said bundled message regarding said series of closed connections to said control processor by said network processor; and_network and end-host efficiency for web communication comprising a method of receiving a request to establish a TCP connection from a client by a network processor in a load balancer of Ramanathan et al., as modified by Gupta et al., and the results would have been predictable to one of ordinary skill in the art, namely, one skilled in the art would have readily recognized a system and method of web server load balancing.

Consider claim 3, as applied to claim 2. Ramanathan et al., as modified by Gupta et al. and Jindal et al., discloses a method wherein said server in said server farm is identified using information extracted from said client's request message ((“In another method of load balancing, specialized hardware is employed to store information concerning the servers hosting instances of a replicated service. In particular, according to this method information is stored on a computer system other than the system that initially receives clients' requests. The stored information helps identify the server having the smallest load (e.g., fewest client requests). Based on that information, a

user's request is routed to the least-loaded server. In a web-browsing environment, for example, when a user's service access request (e.g., a connection request to a particular Uniform Resource Locator (URL) or virtual server name) is received by a server offering Domain Name Services (DNS), the DNS server queries or passes the request to the specialized hardware. Based on the stored information, the user's request is then forwarded to the least-loaded server offering the requested service.") Jindal et al., column 1 line 59 – column 2 line 8).

Consider claim 4, as applied to claim 2. Ramanathan et al., as modified by Gupta et al. and Jindal et al., discloses a method wherein said control message (("Lying at the core of the explosion of the popularity and usage of the Internet is the Web server and browser communication protocol known as hypertext transfer protocol (HTTP). HTTP is the network protocol used to deliver virtually all files and other data, known collectively as resources, on the worldwide Web. These resources include HTML files, image files, query results, etc. This network protocol typically takes place through TCP/IP sockets. As with other network protocols, HTTP utilizes a client-server model. In this model, an HTTP client (such as a consumer) opens a connection and sends a request message to an HTTP server (e.g. a corporate Web server). Once the HTTP server has received the request from the client, it returns a response message, typically containing the resource that was requested by the client. For most typical browsing transactions on the Internet, the server then closes the connection after delivering the response. As such, HTTP is a stateless protocol, i.e. not maintaining any connection information between transactions.

While HTTP 1.1 does maintain persistent connections as a default, HTTP 1.1 also includes a "Connection: close" header that will close the connection after the corresponding response is sent. The actual mechanism of an HTTP transaction, such as a Web browsing connection, may be better understood with reference to FIG. 4, which illustrates the basic request/response message flow between a client and a server. As may be seen from this simplified figure, a client 500 establishes a TCP connection to a server 502 by transmitting a connect request (TCP syn) to the server 502. The server 502 responds to this connect request by transmitting an acknowledgment (TCP syn+ack) to the client 500 who then completes the connect request by acknowledging (TCP ack) the server's acknowledgment of its initial request.") Ramanathan et al., column 1 line 41 – column 2 line 6) comprises information used to enable said network processor to create entries in a forwarding table to ensure packets from said client are transmitted to said server and to ensure packets from said server are transmitted to said client ((“In the illustrated embodiment of the invention, status objects periodically interact with instances of application 104 to collect application-specific statistics that will be used to select a preferred server. For example, if application 104 were a DBMS, a status object may gather the number of database accesses, the number of requests received or pending, etc. for one instance of the application. As another example, if application 104 were an electronic mail program, a status object may periodically gather the number of inbound and/or outbound messages in queue, the number and size of mailboxes, etc. Besides status objects, other computer-readable instructions (e.g., in the form of objects, agents or modules) are also

executed (also described below) to collect, assemble and analyze the various pieces of information provided by the status objects and to update lookup table 102. The objects or agents within a load balancing framework may be created in a suitable programming or script language and then configured and installed on each of servers 110, 112 and 114 and/or on central server 100.") Jindal et al., column 7 lines 11-30).

Consider claim 5, as applied to claim 2. Ramanathan et al., as modified by Gupta et al. and Jindal et al., discloses a method wherein said control message comprises information to establish a TCP connection between said load balancer and said server ((“In another method of load balancing, specialized hardware is employed to store information concerning the servers hosting instances of a replicated service. In particular, according to this method information is stored on a computer system other than the system that initially receives clients' requests. The stored information helps identify the server having the smallest load (e.g., fewest client requests). Based on that information, a user's request is routed to the least-loaded server. In a web-browsing environment, for example, when a user's service access request (e.g., a connection request to a particular Uniform Resource Locator (URL) or virtual server name) is received by a server offering Domain Name Services (DNS), the DNS server queries or passes the request to the specialized hardware. Based on the stored information, the user's request is then forwarded to the least-loaded server offering the requested service.") Jindal et al., column 1 line 59 – column 2 line 8).

Consider claim 7, as applied to claim 2. Ramanathan et al., as modified by Gupta et al. and Jindal et al., discloses a method further comprising the step of: extracting information from said bundled message regarding said series of closed connections by said control processor ((“The server is not the only party to this type of transaction who knows that the connection should be terminated once the request has been satisfied. Indeed, the client also knows that the TCP connection to the server need not be maintained once the client has received its requested resource. Therefore, in an alternate embodiment of the present invention, the client may bundle the HTTP Get request with a Disconnect request. The server is then able to send the requested resource with a send and disconnect IRP indicating to the stack of impending send and disconnection of the connection through the same IRP, while saving overhead of receive path for a special FIN frame. Once this resource has been sent, the server may close the TCP connection without further notification (through FIN frame) from the client (acknowledgelessly closing the connection). This further enhances server performance since the server may close the TCP connection without having to wait for any disconnect frame from the client. The performance enhancements made available through the system of the invention described above all presuppose completion of the IRP on the processor that originated it. While this is a foregone conclusion for single processor servers, conventional multiple-processor servers require a queuing state and an IPI to ensure that all IRPs are so completed as discussed above with reference to FIG. 7. However, this queuing state and subsequent IPI generation creates extra overhead for the server and adversely impacts its performance. Therefore, in multiple-

processor servers employing the system of the instant invention, all IRPs are directly completed on the originating processor without a queuing state as illustrated in FIG. 3. As is illustrated, the processor 300 running thread 302 directly receives the IRP completion 304 without a queuing state as required by conventional systems. Other processors 306 in the server are not required to service this IRP completion 304, and therefore need generate no IPIs. The system of the invention also minimizes the cache line movement incurred conventional systems, which further enhances performance.") Ramanathan et al., column 11 lines 7-45).

Response to Arguments

8. Applicant's arguments filed 05 October 2008 with respect to claims 2-5 and 7 have been considered but are moot in view of the new ground(s) of rejection.

The examiner has cited particular columns and line numbers in the references as applied to the claims above for the convenience of the applicant. Although the specified citations are representative of the teachings in the art and are applied to the specific limitations within the individual claim, other passages and figures may apply as well. It is respectfully requested from the applicant, in preparing the responses, to fully consider each of the cited references in entirety as potentially teaching all or part of the claimed invention, as well as the context of the passage disclosed by the examiner.

Conclusion

9. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL.** See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any response to this Office Action should be faxed to (571) 273-8300 or mailed to:

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Hand-delivered responses should be brought to

Customer Service Window
Randolph Building
401 Dulany Street
Alexandria, VA 22314

Any inquiry concerning this communication or earlier communications from the Examiner should be directed to Mark Fearer whose telephone number is (571) 270-1770. The Examiner can normally be reached on Monday-Thursday from 7:30am to 5:00pm.

If attempts to reach the Examiner by telephone are unsuccessful, the Examiner's supervisor, Tonia Dollinger can be reached on (571) 272-4170. The fax phone number for the organization where this application or proceeding is assigned is (571) 273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free) or 571-272-4100.

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist/customer service whose telephone number is (571) 272-2600.

Mark Fearer
/M.D.F./
December 16, 2008

/Tonia LM Dollinger/

Supervisory Patent Examiner, Art Unit 2443